**LAB ASSIGNMENT -4.3**

**Course: AI Assisted Coding**

**Name: Challa Shreevidya**

**Ht no: 2403A51419**

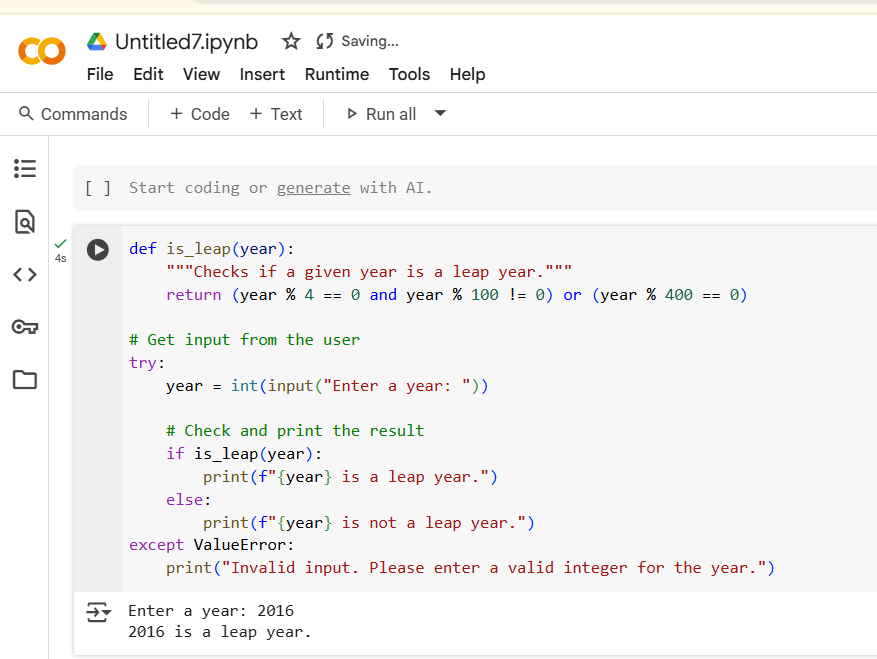
**Batch-16**

**#Task 1:**

**#Prompt:**

write a program in python by taking input from the user, that checks the given year is leap year or not using function.

**#Code & Output:**

****

**#Explanation:**

The Leap Year Checker: We have a special tool (a function called is\_leap) that knows the rules for figuring out if a year is a leap year.

Its rule is: If a year can be divided by 4, but not by 100, it's a leap year.

However, there's an exception: If the year can be divided by 400, it is a leap year, even if it can be divided by 100.

Asking the User: The program asks you to type in a year.

Trying to Understand: It tries to understand the number you typed in.

Using the Tool: It gives the number you typed to the is\_leap tool. The tool checks the rules.

Telling You the Answer: Based on what the is\_leap tool says (yes or no), the program tells you if the year is a leap year or not.

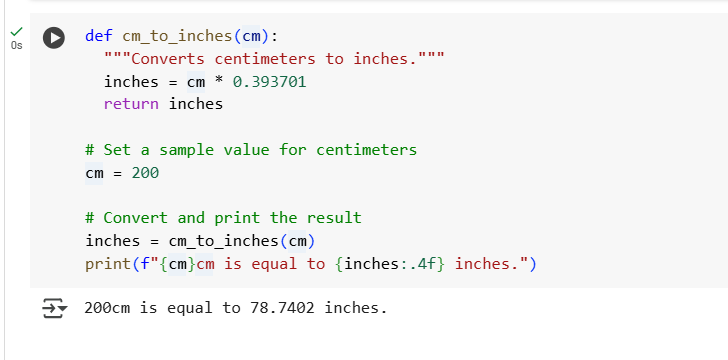
Handling Mistakes: If you type something that's not a number, the program says, "Hey, that's not a valid year!"

**#Task 2:**

**#Prompt:**

write a program in python that converts centimeters to inches using function.(One shot)

**#Code & Output:**

****

**#Explanation:**

The Conversion Machine: We have a little machine (a function called cm\_to\_inches) that knows how to change centimeters into inches.

You give it a number in centimeters (cm).

It does a calculation: it multiplies the centimeters by a special number (0.393701).

It gives you back the answer in inches (inches).

Setting the Starting Point: We decide to start with 200 centimeters. We put cm = 200.

Using the Machine: We use our conversion machine (cm\_to\_inches) and give it the 200 centimeters. The machine figures out how many inches that is.

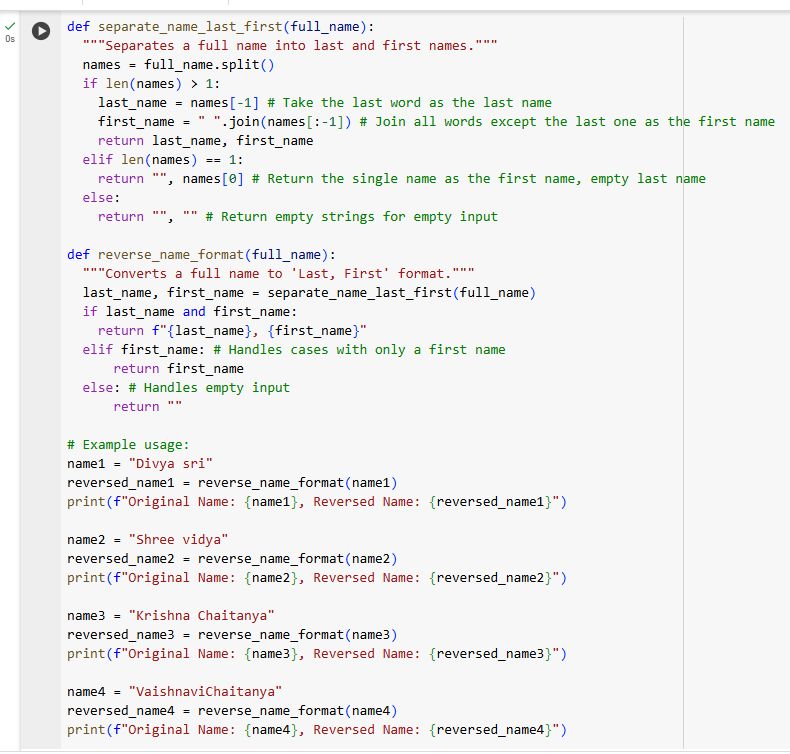
Showing the Answer: Finally, the program tells you the original centimeters and what that equals in inches. It formats the inches number nicely with four decimal places.

**#Task 3:**

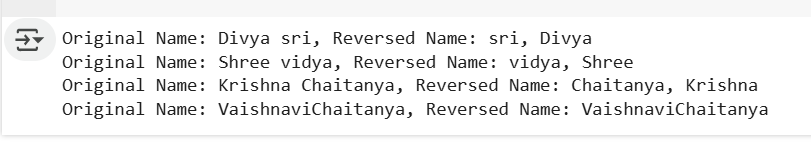
**#Prompt:**

write a program in python that converts the given full name in reverse last,first using function.(few-shot)

**#Code:**

****

**#Output:**

****

**#Explanation:**

1. Helper 1: The Name Splitter (separate\_name\_last\_first)
   * You give it a full name (like "Divya sri").
   * It looks for spaces and breaks the name into separate words ("Divya", "sri").
   * It then figures out which word is the *last* name (usually the last word) and which is the *first* name (the word or words before the last one).
   * It gives you back the last name and the first name separately.
2. Helper 2: The Name Reverser (reverse\_name\_format)
   * You give *this* helper a full name.
   * This helper uses the *first* helper (separate\_name\_last\_first) to get the last name and the first name.
   * Then, it puts them back together in a specific order: Last Name, First Name.
   * If there was only a first name to begin with, it just gives you the first name back.
   * If the input was empty, it gives you nothing back.

The rest of the code just shows examples of how to use the reverse\_name\_format helper with different names and prints the original name and the new "Last, First" version so you can see the result.

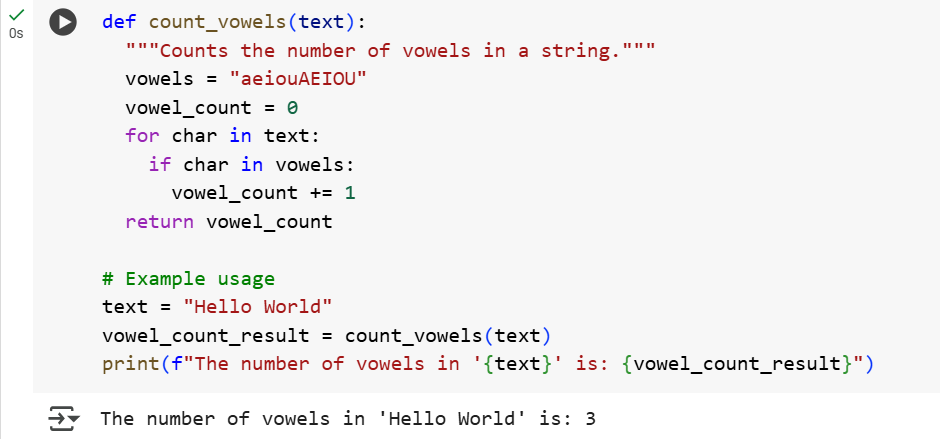
**#Task 4:**

**#Prompt:**

Zero-shot prompt:

write a program in python that counts the number of vowels in a string using functions.(zero-shot)

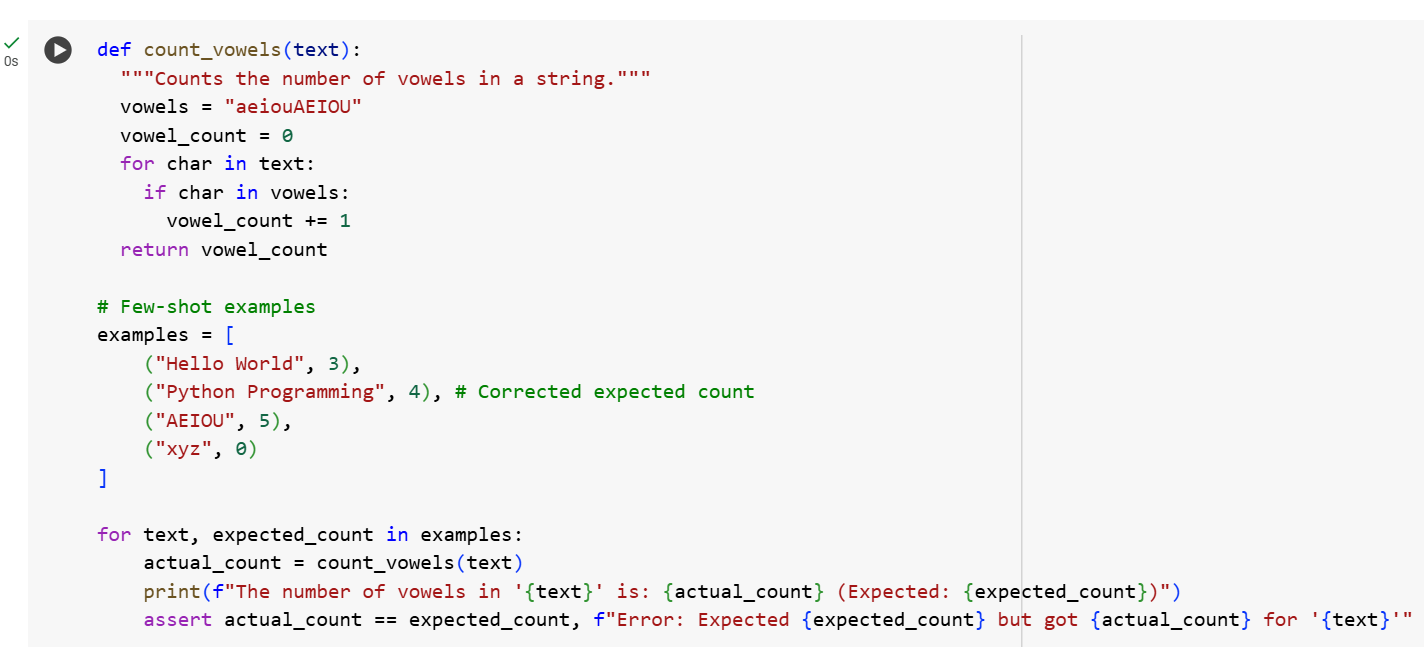
**#Code& Output:**

****

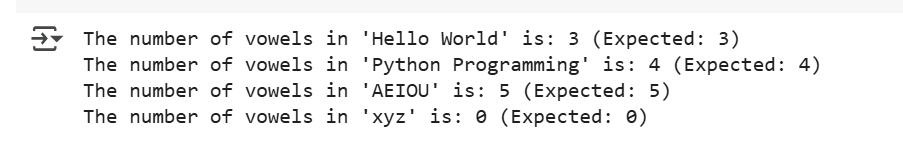
Few-shot prompt:

Write a program in python that counts the number of vowels in a string using functions.(Few-shot)

**#Code:**

****

**#Output:**

****

**#Explanation:**

1. def count\_vowels(text):: This line starts a reusable block of code called a "function" named count\_vowels. It takes one piece of information, which we'll call text (this is the string you want to check).
2. vowels = "aeiouAEIOU": This creates a list of all the letters that are considered vowels (both lowercase and uppercase).
3. vowel\_count = 0: This sets up a counter called vowel\_count and starts it at zero. This counter will keep track of how many vowels we find.
4. for char in text:: This is a loop that goes through each character, one by one, in the text you provided. For example, if the text is "Hello", the loop will first look at 'H', then 'e', then 'l', and so on.
5. if char in vowels:: Inside the loop, for each char (character), this line checks if that character is present in our list of vowels.
6. vowel\_count += 1: If the character IS a vowel, this line adds 1 to our vowel\_count.
7. return vowel\_count: After the loop has checked every character in the text, this line gives back the final total of vowel\_count.

The part below the function is just an example of how to use it:

* text = "Hello World": This sets the variable text to the string "Hello World".
* vowel\_count\_result = count\_vowels(text): This calls the count\_vowels function with "Hello World" as the input and stores the result (the vowel count) in a variable called vowel\_count\_result.

**#Comparision :**

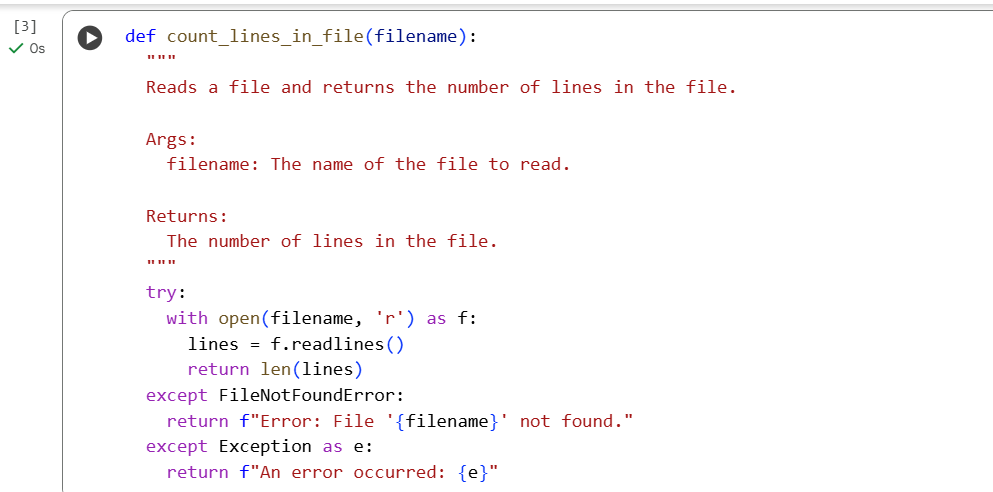
* Both the zero-shot and few-shot prompts successfully generated Python code that correctly counts vowels in a string.
* The difference in the output of the two programs is due to the use of different sample strings for each program, not a difference in the vowel counting logic.
* To truly compare the performance or output of the generated code itself, the same input string should be used for both programs.

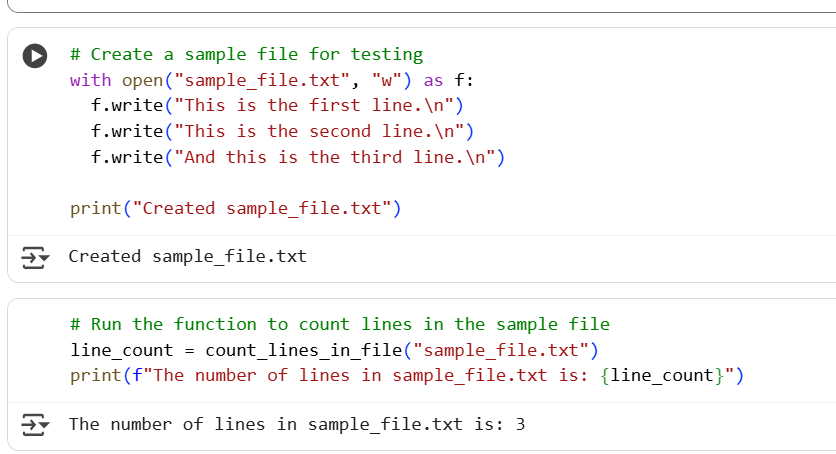
**#Task -5:**

**#Prompt:**

write a python function that takes a filename as input,reads the file, and returns the number of lines in the file.

**#Code & Output:**





**#Explanation:**

First, we defined a Python function called count\_lines\_in\_file. This function is designed to open a file, read all its lines, and then tell you how many lines there are. It also includes error handling in case the file isn't found or another issue occurs.

Then, you created a sample file named sample\_file.txt and wrote a few lines into it. This was just to have a file to test the function on.

Finally, you called the count\_lines\_in\_file function, passing the name "sample\_file.txt" to it. The function then read the sample file, counted the lines, and the result (which was 3) was stored in a variable called line\_count and then printed out.